Data structures and Algothrims

10/08/22

complexity Analysis

complexity → Externally → Processor, Memory size, H/W settings
complexity → Internally → Time, space

Analysis → Experimental
Analysis → Theortical

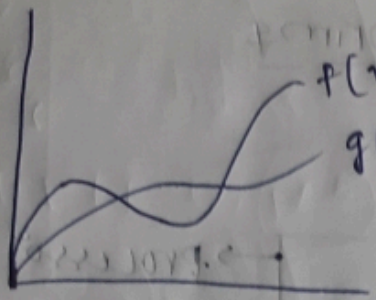
Time space trade of → only one is good statisfied

Q) (Asympotic Notation Explain?) – (10m)

Asympotic Notation

- Omega Notation (Ω) → Lowerbounded values → Best case (minimum no.of time)
- Theta Notation (θ) → Tight bounded values → Average case
- Big-O Notation (O) → upperbounded values → Worst case

Omega Notation

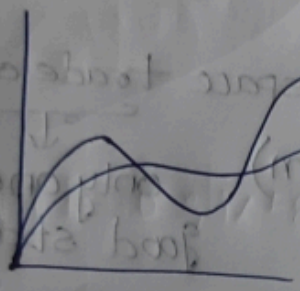


$f(n) = g(n)$

$\Rightarrow \boxed{f(n) \geq c\,g(n)}$

c → constant

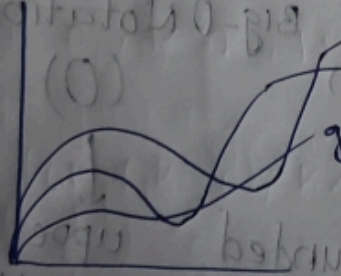
Big-O Notation



$f(n) = g(n)$

$\Rightarrow \boxed{f(n) \leq c\,g(n)}$

Theta Notation



$f(n), g_1(n), g_2(n), c_1, c_2$

$f(n) = g(n)$

$\Rightarrow \boxed{c_1 \cdot g_1(n) \leq f(n) \leq c_2 \cdot g_2(n)}$

Rules of Big-O Notation to find time complexity

1. Nested loops are multiplied together
2. Sequencial loops are added
3. Larger term will kept and smaller term will drop.
4. conditional checks are constant.
5. constants are dropped.

---

1. (Nested loop)

```
for(i=0; i<n; i++)        → O(n)
{
   for(j=0; j<n; j++)     → O(n)
   {
     - - - -
   }
}
```

(i<n, j<n → very imp)

complixity is $O(n^2)$

bcz → Rule ①

2. (sequential loops)

```
for (i=0; i<n; i++)  → O(n)
{
   ==
}
for (j=0; j<n; j++)  → O(n)
{
   ==
}
```

compixity is $O(n)+O(n)$

$= O(2n)$

bcz → Rule → ②

finally complixity is $O(n)$

⇒ (2 is dropped bcoz it is constant → Rule 5)

3.

3) for(i=0 ; i<58 ; i++)

{

= =

}

for (j=0; j<5 ; j++)

{

= =

}

these condition checks are constants so time complexity becomes

O(1)

Rule (4)

(58, 5)

4) for (i= 0 ; i<58 ; i++)

{

}

] O(1)

for (j=0 ; j<n ; j++)

{

for (k=0 ; k<n ; k++)

{

= =

}

}

] $O(n^2)$

for (v=0 ; v<n ; v++)

{

}

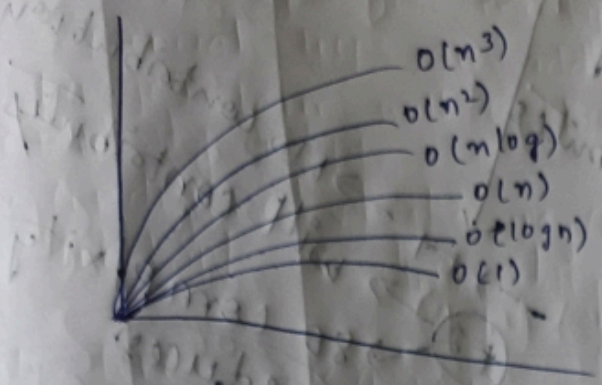
] O(n)

Complexity is $O(n^2)$ by Rule (3)

i.e Larger kept smaller leave

(imp)

Graph



(Asmpotic Ans Ends)

(10m)

① for (int i=0; i<n; i++) → O(n)
{
  for (j=0; j<n; j++) → O(n) ⇒ O(n²)
  {

  }
}

② for (i=0; i<100000; i++) → O(1)
{

}
(constant)

③ for (i=0; i<n; i++) → O(n)
{

}

Ay for (i=0; i<n; i++) → O(n)
{
  for (j=0; j<m; j++) → O(m)
  {

  }
} ⇒ O(nm)

imp
for (i=0; i<n; i++) → O(n)
{
  for (j=0; j<i; j++) → O((n+1)/2)
  {
    --
  }
}
b/c it depends on ① outerloop

① n×(n+1)/2 ∴ count
n²/2 + n/2 count
O(n²)

6) imp Q for (int i=0; i<n; i*=2)
{

}

i=0 → No contribution in complity
i<n → contributes
i*=2 →
--, i++ → No contribution
*2 → will contribute so complxity reduces → bcoz see graph

Complixity is **O(logn)**

O(logn)

O(n) > O(logn)

7) for (i=0; i<n; i++) → O(n)
{
------
for (j=0; j<n; j*=2) ↳ O(logn)
{

}
}

complixity

O(nlogn)

(8) $P_1 \rightarrow 3n^4 + 2n^3 + 4n + 5n + 2 + 1$

$P_2 \rightarrow 2n + 1$

$P_3 \rightarrow 15n^2 + 4n - 1$

$\rightarrow O(n^4)$

$O(n)$

$O(n^2)$

For polynomials complixity is highest degree complexity is $\Rightarrow O(n^4)$

11/08/2022

Array

* Length (No. of Elements in array) = UB - LB + 1

Ex :- 1) A (-5, 10) $\Rightarrow 10 - (-5) + 1$

LB UB

$\Rightarrow 10 + 6 = 16$ elements present.

* Representation of Linear Array in Memory :-

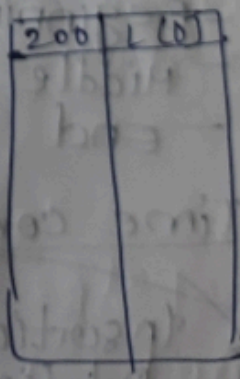
LOC[LA[K]] = Base(LA) + W(K - LB)

W $\Rightarrow$ Size of datatype.

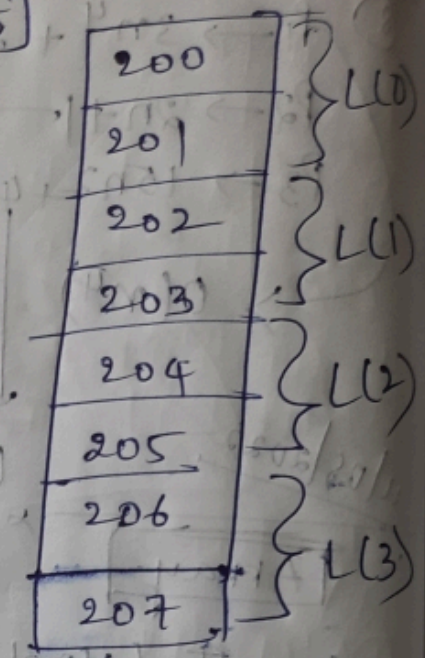
Ex :- Address at LOC $\rightarrow$ (6)

Ans's LOC(LA(6)) = $200 + 1(6 - 0) = 200 + 6 = 206$

Q :- Find the address for LA[6] Each element of the array occupy 1 byte



② Q Find the address for LA[15] Each element of the array occupy 2 bytes?



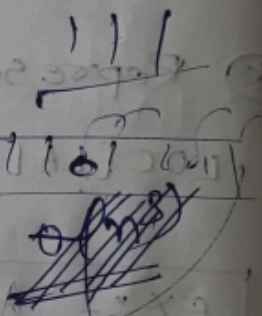
Sol:- LA(15) = 200 + (2)×(15−0)
= 200 + 30
LA(15) = 230

⊛ Algorithm for traversing an array.

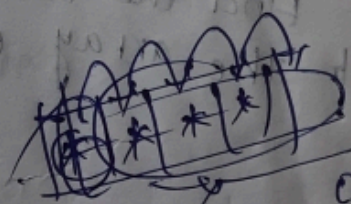
17/August.

⊛ Insertion and Deleting

⊛ Insertion: Adding an element.
- Beginning
- Middle
- End



⊛ Deletion: Removing an element.
- Beginning
- Middle
- End



→ Time complixity O(n)

⇒ Insertion Deletion at Beg → O(1)

⇒ Insertion Deletion at End → O(n)

If array is Empty

Insertion at beginning = O(n) => If array is filled fully
=> Deletion

=> Insertion at middle = O(n)
Deletion

(*) Linear search

Att — 5M
CA — 35M —> Hackerrank platform
ETerm (MCQ + subjective) —> 40M

CA Mode

1) Practice problem —> 30M
2) coding Test — 15M (After Mid)
3) online Quiz — 10M

—> 15th October 2022 —> in HackerRank.

—> Best 1 out 2

QUIZ 1 —> 10th September
QUIZ 2 —> 5th November